New Single Asymmetric Error Correcting Codes

by

Sultan Ahmad Muhammad Al-Muhammadi

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER SCIENCE

June, 1998

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

New Single Asymmetric Error Correcting Codes

BY

Sultan Ahmad Muhammad Al-Muhammadi

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

Computer Science

June 1998

KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS
DHAHRAN, SAUDI ARABIA

COLLEGE OF GRADUATE STUDIES

This thesis, written by **Sultan Ahmad Muhammad Al-Muhammadi** under the direction of his Thesis Advisor and approved by his Thesis Committee, has been presented to and accepted by the Dean of the College of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE** in **COMPUTER SCIENCE.**

THESIS COMMITTEE

١٥/٦/٩٨

Dr. Sulaiman Al-Bassam (Chairman)

Dr. Saud Al-Semari (Member)

Dr. Muslim Bozyigit (Member)

6/15/98

Department Chairman

Dean, College of Graduate Studies

Date: 15/6/98

كلية الدراسات العليا
COLLEGE OF GRADUATE STUDIES

To My Beloved Mother

Acknowledgment

First and foremost, all praise is due to Allah, the Almighty, Who gave me every reason to carry out this work. I thank Him for the unlimited favors that He bestowed on me, and seek His mercy and forgiveness.

Acknowledgment is due to *King Fahd University of Petroleum & Minerals* for its support of this research.

I would like to express my deep appreciation to *Dr. Sulaiman Al-Bassam*, thesis advisor, for his patient guidance and his generous support for this research. I would also like to express my profound gratitude to my other committee members, *Dr. Saud Al-Semari* and *Dr. Muslim Bozyigit*, for their valuable suggestions and for the innumerable improvements in the presentation.

I would like to thank the chairman, the faculty and the staff of the *Information and Computer Science* Department for their great support. I am fortunate to have had the benefit of comments from my roommate, *Muhammad Al-Sahli*, who reviewed a previous draft of this thesis. Special thanks go to *Farooq Ashraf* for his great assistance in using LaTeX, and to *Ahmad, Mutlaq, Garout, Husni, Hatim, Khalid, Adnan*, and my officemate, *Adel*, for their encouragement and good wishes.

Finally, I would like to express my sincere gratitude to my mother and my family for their constant encouragement and moral support.

Contents

List of Tables

List of Figures

Thesis Abstract

Name: Sultan Ahmad Muhammad Al-Muhammadi
Title: New Single Asymmetric Error Correcting Codes
Major Field: Computer Science
Date of Degree: June, 1998

Research in the area of asymmetric errors started the in late 1950's. Extensive research has been developed, since then, on the theory of asymmetric error-control codes. The interest of the asymmetric error-control codes has become increasingly apparent due to some new applications that assume asymmetric channels for communication. Failures in such channels normally result in asymmetric errors. This thesis presents new single asymmetric error correcting codes. These new codes have higher information rate than the existing codes for any code length greater than 10, except 12 and 15.

It is known that, for a given length n, a code of size $\lceil 2^n/(n+1) \rceil$ can be obtained from the Abelian group partitioning of all the 2^n binary code words. This thesis introduces codes of sizes greater than $\lceil 2^n/n \rceil$ code words for many values of n. For $n = 17$, the size of the introduced code is 2^{13}, whcih is equal to $\lceil 2^n/(n-1) \rceil$. So, it is now possible to encode the 13-bit messages into a single asymmetric error correcting code of length 17, i.e. with only four bits of redundancy. The construction method of the proposed codes is based on the Cartesian product of two sets of partitioned codes of smaller lengths. Two algorithms for finding good partitions are discussed. The first one is based on the Cartesian product method itself. The other is based on graph-coloring techniques. Using these algorithms, some useful partitions for the construction method were obtained.

Master of Science Degree
King Fahd University of Petroleum and Minerals
Dhahran, Saudi Arabia
June, 1998

خلاصة الرسالة

الاسم: سلطان أحمد محمد الخمدي

عنوان الرسالة: شفرات جديدة لتصحيح الخطأ الغير متماثل

التخصص: علم الحاسب الآلي

تاريخ الشهادة: صفر ١٤١٩هـ

بدأت دراسة الأخطاء الغير متماثلة في أواخر الخمسينات، حيث أجريت أبحاث عديدة حول نظرية الشفرات المتحكمة بالأخطاء الغير متماثلة. وظهرت تطبيقات جديدة يتم فيها الاتصال باستعمال نوع خاص من القنوات يعرف باسم (القناة الغير متماثلة)، حيث يفترض في هذا النوع من القنوات حدوث أخطاء غير متماثلة فقط. ونتيجة لظهور هذه التطبيقات وانتشارها، انصرف المزيد من البحث والاهتمام نحو الشفرات المتحكمة بالأخطاء الغير متماثلة. وما هذه الرسالة إلا امتداد للجهود المبذولة في هذا المجال، فهي تقدم شفرات جديدة لتصحيح الخطأ (المنفرد) الغير متماثل تفوق الشفرات الموجودة من قبل في نسبة توفير المعلومات. ويتجلى هذا التفوق في الشفرات التي يزيد طولها على عشرة رموز ثنائية (binary bits) باستثناء الشفرة ذات الاثني عشر رمز ثنائيا والشفرة ذات الخمسة عشر رمزا ثنائيا.

من المعلوم أنه بالإمكان الحصول على شفرة تحتوي على $2^{n-(n-1)}$ كلمة مشفرة، حيث n طول الشفرة، وذلك عن طريق التجزيء **الأبليني** للزمرة التي تحوي جميع المتتاليات الثنائية المشفرة ذات الطول n. ولكن هذه الرسالة تقدم شفرات تحوي أكثر من $\lceil 2^{n-\lceil 2\log n \rceil} \rceil$ كلمة مشفرة لمعظم قيم n. والجدير بالذكر أن الشفرة ذات السبعة عشر رمزا ثنائيا المطروحة هنا تحوي 2^{13} كلمة مشفرة، أي ما يعادل $\lceil 2^{n-(n-1)} \rceil$. وهذا يعني أنه بالإمكان تشفير جميع الرسائل ذات الطول ١٣ بشفرة طولها ١٧ رمزا ثنائيا قادرة على تصحيح الخطأ الغير متماثل بإضافة أربعة رموز ثنائية فقط. كما تقدم هذه الرسالة طريقة إنشاء الشفرات المطروحة، وهي مبنية على الجداء الديكارتي لمجموعتين من الشفرات القصيرة المجزأة. وتقدم أيضا خوارزميتين لإيجاد تجزيئات جيدة تعزز طريقة الإنشاء. فأما الخوارزمية الأولى فمبنية على طريقة الجداء الديكارتي نفسها، وأما الأخرى فمبنية على تلوين الأشكال. وبنهاتين الخوارزميتين تم الحصول على تجزيئات مفيدة لزيادة الكلمات المشفرة الناتجة عن طريقة الإنشاء.

درجة الماجستير في العلوم

جامعة الملك فهد للبترول والمعادن

الظهران – المملكة العربية السعودية

صفر ١٤١٩هـ

Chapter 1

Introduction

Computer data should always remain correct in all sorts of processing, i.e. when it is written into memory or output device, stored, read from memory or input device, communicated, or manipulated. The growing complexity of new computers makes it very impractical to depend on reliability of components and devices for reliable operations. Some redundancy is needed to ensure the detection and/or correction of errors which will invariably occur as information is being stored, transferred, or manipulated.

An extensive theory of error-control coding has been developed since the 1950's [2], [8], [11]. The primary emphasis of this theory has been the design of reliable communication systems. The problem of reliable computation differs significantly from the problem of reliable communication. For example, communication error-control schemes usually assume a perfectly reliable computing and processing at the

transmitter and receiver, and have less severe restraints on computation time for error correction. In addition, they are subject to different statistics of error occurrences than those that occur in computer systems. The principles that have been discovered by communication coding theorists are so fundamental that they also are basic to the understanding and design of error control for reliable computation.

In this thesis, new error-control codes are proposed. These codes are capable of correcting a single asymmetric error. The class of asymmetric errors, introduced in section 2.1, was considered recently in the theory of error control coding. In the past three decades, most of the research took place under the assumption of symmetric errors. Unlike symmetric errors, the issues of asymmetric errors have not been well studied in literature yet in spite of the extensive research done so far. This thesis is an extension of the research done in the area of asymmetric errors.

The basic idea of the construction method of the proposed codes is to form the code from the Cartesian product of two sets of smaller partitioned codes. The method is quite sensitive to the sizes of the smaller partitions. Indeed, the better partitions used in this method, the more code words constructed. Therefore, the issue of obtaining better partitions is considered in this thesis and useful partitions of binary vectors are presented.

1.1 Statement of Problem

The main consideration of this thesis is the following problem.

Problem: *"Given a code length, what is the maximum possible size of a code capable of correcting a single asymmetric error?"*

In this thesis, codes of lengths from 2 to 22 are considered. These lengths are commonly practiced in real applications. The construction of the codes is given here as well. Codes of lengths greater than 22 can be easily constructed using the same method. The size of a constructed code of length n gives a lower bound for this problem, i.e. a lower bound on the maximum possible size of a code of length n that is capable of correcting a single asymmetric error.

1.2 Thesis Outlines

The outline of this thesis is summarized as follows:

1. Define the Asymmetric Error Correcting Codes.
2. Review the best known single asymmetric error correcting codes.
3. Propose a new method to construct single asymmetric error correcting codes.
4. Introduce new single asymmetric error correcting codes using the proposed method.

1.3 Summary of Results

The followings are the main results of this work:

- The proposed codes are better than existing codes when the code length (dimension) n is greater than 10, except for $n = 12$ and $n = 15$.

- The proposed codes for $n = 12$ and $n = 15$ are also the best known codes that can be explicitly constructed, since the existing codes for these values of n are based on combinatorial arguments.

- In many cases, the sizes of the proposed codes are greater than $\lceil 2^n/n \rceil$, where n is the length of the code.

- The size of the proposed code of length $n = 17$ is equal to $8192 = 2^{13}$ code words, which means that four bits of redundancy are enough to encode all the 13-bit messages into a single asymmetric error correcting code of length 17.

- The construction method of the proposed code is based on the Cartesian product of two sets of partitioned codes of smaller dimenssions.

- The proposed method in turn is used to obtain good partitions for binary vectors. Graph-coloring is also used for the same purpose.

1.4 Thesis Organization

The thesis is organized as follows: In Chapter 2, the asymmetric error correcting code is highlighted. Previous methods of finding or constructing asymmetric error correcting codes are reviewed. A summary of the best known single asymmetric error correcting codes is given. In Chapter 3, new single asymmetric error correcting codes are proposed and their construction method is explained. The size and the dimension of the codes constructed by this method are discussed. Finally, in Chapter 4, the main contributions of this thesis are summarized and possible future research directions are outlined.

Chapter 2

Asymmetric Error Correcting Codes: An Overview

2.1 Introduction

Most errors in computer systems are caused by faulty components of the system. In the absence of faults, errors may still occur due to random disturbances or noise. Such errors are rare except in long distance communication. The types of error statistics which occur in data channels, and in memory, logic, and arithmetic circuits are many and varied. For example, the errors can be categorized as *independent bit errors* and *physically clustered bit errors*. If the bits of a coded word are stored in physically remote places, their errors are more likely to be independent, whereas if the bits of a word are located physically close, there is a greater chance that errors

will be clustered.

Another classification of errors is to categorize them as *symmetric* and *asymmetric* errors. In case of symmetric errors, both $0 \rightarrow 1$ and $1 \rightarrow 0$ errors occur with almost the same probability. While in the case of asymmetric errors, a given word or operational unit suffers only $1 \rightarrow 0$ or only $0 \rightarrow 1$ errors. The errors in magnetic tapes and some of the random access memories (RAMs) can be considered as symmetric errors. On the other hand, the failure in the memory cells of some of the LSI single-transistor-cell memories and metal-nitride-oxide semiconductors (MNOS) memories are most likely caused by the leakage of charge. If we represent the presence of charge in a cell by 1 and the absence of charge by 0, the errors in these types of memories can be modeled as $(1 \rightarrow 0)$-type asymmetric errors. This is because the charge cannot be created except by a rewrite process, and hence $(0 \rightarrow 1)$-type errors in the memory cells are almost impossible [12], [13].

Another example of asymmetric errors is the errors in asymmetric channels, or Z-channels, as shown in Figure 2.1. In a Z-channel, the probability of converting a 1 into a 0 ($1 \rightarrow 0$ error) is equal to p, $(0 < p < 1)$, while the probability of converting a 0 into a 1 ($0 \rightarrow 1$ error) can be neglected. There are many applications where such a channel is involved. For example, in fiber optical transmission, the transmitted binary sequence may suffer errors of the type $(1 \rightarrow 0)$ which means a loss of light signal, but very rarely a $(0 \rightarrow 1)$-type error occurs which means creation of light.

Beside symmetric and asymmetric error classes, another class in this classification

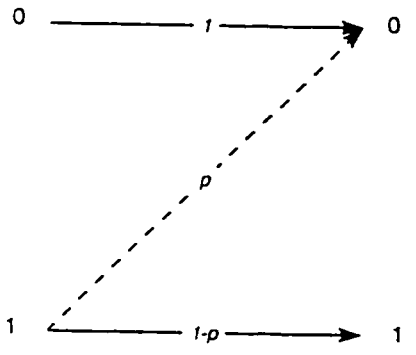


Figure 2.1: The asymmetric channel (Z-channel).

is the *unidirectional* errors, where a given word has only $1 \rightarrow 0$ or only $0 \rightarrow 1$ errors but the decoder does not know a priori the type of the error. To avoid any confusion in classifying asymmetric and unidirectional errors, let us show the difference between these two classes. In considering situations where all errors in a coded block are of one type (i.e. all $1 \rightarrow 0$ or all $0 \rightarrow 1$), two different cases are distinguished. The errors are called asymmetric if only one of the two types of errors can occur and the type of the error is known a priori. The errors are called unidirectional if all the errors in any given code block are of one type (either all $1 \rightarrow 0$ or all $0 \rightarrow 1$), but it is not known a priori which of the two types has occurred. The various faults in many LSI/VLSI devices, such as a failure in the power supply, usually result in a unidirectional error [12].

After understanding the class of asymmetric errors, the difference between error correcting and error detecting codes will be clarified in Section 2.2. Then, in Section 2.3, some terminologies and definitions are given as well as the main theorem of

the asymmetric error correcting codes. Although the issue of asymmetric errors is relatively new, compared with the symmetric errors, extensive research has been done in this area since the late 1950's. In Section 2.4 a brief history of the previous work on asymmetric error correcting codes is given, and the best known codes are summarized. Finally, Section 2.5 summarizes the main points of Chapter 2.

2.2 Error Correcting Codes

Before giving the formal definition of the asymmetric error correcting codes, let us explain an important consideration which is the relative use of error detection and error correction. Error detection normally results in an interruption in the computation, followed by possibly a retry of some or all of the past computations, or possibly a switching from a supposedly faulty part of the computer to a presumed reliable part. Sometimes, this interruption is followed by some maintenance procedure. However, error correction permits the computation process to continue uninterrupted. This would seem to favor error correction over error detection, but there are some mitigating factors. For a given amount of redundancy, if the number of error patterns which the decoder attempts to correct is increased, the probability of an undetected error increases; and the computational decoding complexity increases rapidly as the amount of error-correction capability is increased

Error correction is more important in computing than in communication. A sat-

isfactory method of obtaining extremely reliable communication in the presence of information errors can be provided by an error detection combined with acknowledgment and retransmission protocols[1]. While, in computing, if the presence of errors is detected in a word retrieved from memory, there may be no way of determining what the correct word is unless some error correcting techniques are used [9].

2.3 Terminologies and Definitions

Considering single asymmetric error correcting codes in this thesis, we assume an ideal asymmetric binary channel, as shown in Figure 2.1. Also, we assume that all asymmetric errors are of the type $(1 \rightarrow 0)$ error, and that there is at most one error per code word. Since the probability of having one error is very small, the probability of having two errors can be neglected. The coded binary vector is called a *code word*. A *single asymmetric error correcting code* is a set of code words of some length (or *dimension*) n capable of correcting at most one asymmetric error encountered while transmitting a code word through an ideal asymmetric binary channel.

Before giving the main theorem in this chapter, consider the following definitions.

Definition 2.1 Let $x = [x_1\ x_2\ \cdots\ x_n]$ be a binary vector of length n, where

[1] Not true for one-way communications, like: satellite broadcasting, digital TVs and pagers.

$x_i \in \{0, 1\}$ is the i^{th} bit of x. The *weight* of x is defined by

$$w(x) = |\{i : x_i = 1\}|$$

i.e. the number of 1's in x.

Definition 2.2 Let $x = [x_1\ x_2\ \cdots\ x_n]$ and $y = [y_1\ y_2\ \cdots\ y_n]$ be two binary vectors of the same length. The *asymmetric distance* of x and y is defined by

$$d_a(x, y) = max\{N(x, y), N(y, x)\}$$

where $N(x, y) = |\{i : x_i = 1 \text{ and } y_i = 0\}|$, i.e. the number of bit positions where x has a 1 and y has a 0.

Definition 2.3 The *minimum asymmetric distance* of a code C is defined by

$$\Delta(C) = min\{d_a(x, y) : x, y \in C \text{ and } x \neq y\}$$

Definition 2.4 The *Hamming distance* of two binary vectors, x and y, of the same length is defined by

$$d_H(x, y) = N(x, y) + N(y, x)$$

where $N(x, y)$ is as defined in Definition 2.2.

Definition 2.5 The *minimum Hamming distance* of a code C is defined by

$$H(C) = min\{d_H(x, y) : x, y \in C \text{ and } x \neq y\}$$

The asymmetric distance of two vectors, x and y, is related to the Hamming distance by [19]

$$2d_A(x,y) = d_H(x,y) + |w(x) - w(y)| \tag{2.1}$$

The relation between the asymmetric distance and number of asymmetric errors, which can be corrected, is given in Theorem 2.1 [14]. This is the main theorem in this chapter.

Theorem 2.1 *Any binary code C of asymmetric distance Δ can correct $\Delta - 1$ or fewer asymmetric errors.* It is therefore called a $(\Delta - 1)$ asymmetric error correcting code.

Proof: Without loss of generality, we can assume that asymmetric errors are of the type $(1 \rightarrow 0)$. For any code word $x \in C$, let S_x denote the set of vectors obtained from x by introducing t errors of the type $(1 \rightarrow 0)$, for $0 \leq t \leq \Delta - 1$. Consider two code words $x, y \in C$: since $d_a(x,y) \geq \Delta$, without loss of generality we can assume $N(x,y) \geq \Delta$. Clearly, y cannot become x by any number of $(1 \rightarrow 0)$ errors. Also, $\Delta - 1$ or fewer $(1 \rightarrow 0)$ errors cannot take x to y or into S_y. That is, S_x and S_y are disjoint and the code can correct up to $\Delta - 1$ asymmetric errors.

□

From the above theorem, any code C with $\Delta(C) \geq 2$ is capable of correcting a single asymmetric error. Hence, the goal is to find, for a given dimension n, the largest possible code C_n of length n such that $\Delta(C_n) \geq 2$.

2.4 History and Previous Work

The theory and construction of asymmetric codes have been studied since the late 1950's. In 1959, Kim and Freiman proposed a construction method of asymmetric codes using "prefix/suffix" constructions of codes [6]. The construction of a code of a given length gives a lower bound for the maximum size of all codes of the given length. In 1964, Varshamov gave an explicit upper bound for asymmetric codes[16]. In 1971, Goldbaum obtained tighter upper bounds using integer programming techniques. Two years later, Varshamov used algebraic group theory to construct codes for correcting asymmetric errors [17]. This method is explained in Section 3.2.1. In 1979, the same method was improved and used to construct codes for asymmetric channel by Constantin and Rao [4].

In 1981, Kløve [7] and Delsarte and Piret [5] improved the upper bounds that was obtained by Goldbaum in 1971 by adding more constraints to the integer programming technique. Delsarte and Piret also introduced the "expurgating/puncturing" construction method for asymmetric error correcting codes. They used the idea of constructing a code of length n and asymmetric distance Δ by modifying an initial code with good (Hamming) distance properties by successive judicious deletions of coordinates and vectors [5]. A year later, in 1982, Shiozaki presented a construction method of a t-fold asymmetric error-correcting code of length $n-1$ by expurgating and puncturing any t-fold symmetric error-correcting code of length n.

In 1987, Weber, De Vroedt, and Boekee proposed new upper bounds on the size of asymmetric error correcting codes by further enhancing the constraints for the integer programming technique [18]. A year later, they improved the upper bounds and proposed constructions for asymmetric error correcting codes using a general "expurgating/puncturing" construction method [19]. This method includes, as special cases, the construction method of Shiozaki and some of the constructions of Delsarte and Piret. They proposed a construction method for a code C of length $(n-m)$, where $1 \leq m < n$, and asymmetric distance $\Delta \geq t+1$ which consists of expurgating and puncturing a code C' of length n and Hamming distance $H \geq 2t+1$.

Table 2.1 summarizes the upper and lower bounds for the single asymmetric error correcting codes obtained in the period from 1959 to 1988. All upper bounds in this table were obtained using the integer programming techniques as described in [18]. It is important to mention that the lower bounds in this table are for *constructable* codes, i.e. codes that can actually be constructed.

In 1992, Zhang and Xia derived new lower bounds for asymmetric single-error-correcting codes. The codes were obtained by puncturing constant weight codes and by using a random coding argument. Table 2.2 shows the lower bounds for the codes obtained by Zhang and Xia for lengths from 12 to 19. Although their method is non-constructive, they used probability and counting techniques to show that the asymmetric single-error-correcting codes of the following sizes exist [20].

Table 2.3 summarizes the lower bounds for the best existing single asymmetric

n	lower bounds	upper bounds
5	6^a	6
6	12^b	12
7	18^d	18
8	36^c	36
9	62^d	62
10	108^c	117
11	174^c	210
12	316^a	410
13	586^a	786
14	1096^a	1500
15	2048^a	2828
16	3856^a	5430
17	7296^d	10374
18	13798^a	19898
19	26216^a	38008
20	49940^a	73174
21	95326^a	140798
22	182362^a	271953

Table 2.1: Summary of previous bounds for $\Delta = 2$ (1959-1988).
The upper bounds are obtained by integer programming techniques [18].
The lower bounds are for the following constructable codes:

(a) Code by Varshamov [17].
(b) Code by Kim and Freiman [6].
(c) Code by Delsare and Piret [5].
(d) Code by Constantin and Rao [4].

n	code size
12	340
13	624
14	1139
15	2216
16	4168
17	7688
18	13951
19	26265

Table 2.2: Codes by Zhang and Xia (1992).

error correcting codes. The numbers in parentheses are for constructable codes when the best known bound is non-constructable. In Chapter 3, new codes of better sizes are proposed for values of $n > 10$, except for $n = 12$ and $n = 15$. However, the proposed codes are constructable and they are better than the best known constructable codes for $n > 10$.

2.5 Remarks

The issue of asymmetric errors is relatively new, compared with the symmetric errors. The idea of asymmetric errors comes from some applications like those which use the Z-channel, Figure 2.1, for communication. Examples of these applications include: optical fiber transmission, some of the LSI single-transistor-cell memories and metal-nitride-oxide semiconductor (MNOS) memories. According to their nature, a failure in these applications results in asymmetric errors [12]. The theory of

n	existing code size
2	2^a
3	2^a
4	4^a
5	6^a
6	12^b
7	18^d
8	36^c
9	62^d
10	108^c
11	174^c
12	$340^e(316^a)$
13	$624^e(586^a)$
14	$1139^e(1096^a)$
15	$2216^e(2048^a)$
16	$4168^e(3856^a)$
17	$7688^e(7296^d)$
18	$13951^e(13798^a)$
19	$26265^e(26216^a)$
20	49940^a
21	95326^a
22	182362^a

Table 2.3: Best existing single asymmetric error correcting codes.

(a) Code by Varshamov [17].
(b) Code by Kim and Freiman [6].
(c) Code by Delsare and Piret [5].
(d) Code by Constantin and Rao [4].
(e) Code by Zhang and Xia [20], non constructable.

asymmetric error correcting/detecting codes has not been well established yet. Since the 1950's, much research has been done to present codes for correcting/detecting asymmetric errors. Some construction methods of these codes have been given [19]. In addition, some theorems are stated on the asymmetric error correcting codes and the asymmetric distance of codes [9], [14].

This thesis proposes codes capable of correcting a single asymmetric error per code word. Also, it gives a procedural construction method for the proposed codes (Section 3.2). This method is based on the Cartesian product of two sets of partitioned codes of smaller dimensions. The proposed codes are better than the existing constructable codes for dimensions greater than 10. The construction method is sensitive to the two sets of partitioned codes that are used in the Cartesian product. The better partitioned codes used in this method, the larger codes can be constructed. This thesis, also, presents some useful techniques to obtain "good" partitions for this method (Section 3.3).

Chapter 3

New Single Asymmetric Error Correcting Codes

3.1 Introduction

In this chapter, new codes capable of correcting a single asymmetric error are proposed. The construction of the codes is also given. As a quick reference, Table 3.1 lists the new codes and the best known codes which are mentioned in Section 2.4.

The Construction method of any code is an important issue for its applications. Assume we can prove that a code of a given size does exist, then we may not be able to use it unless we know how to construct it. So, code construction via a procedure is more practical than just showing that a code of a given size does exist. If the construction method is known, one can construct and use the code. Preferably,

n	proposed codes	existing codes
2	2	2^a
3	2	2^a
4	4	4^a
5	6	6^a
6	12	12^b
7	16	18^d
8	28	36^c
9	52	62^d
10	104	108^c
11	180*	174^c
12	336	340^e
13	652*	624^e
14	1204*	1139^e
15	2214	2216^e
16	4232*	4168^e
17	8192*	7688^e
18	14624*	13951^c
19	28548*	26265^e
20	53856*	49940^a
21	101576*	95326^a
22	195700*	82362^a

Table 3.1: New single asymmetric error correcting codes.

(a) Code by Varshamov [17].
(b) Code by Kim and Freiman [6].
(c) Code by Delsare and Piret [5].
(d) Code by Constantin and Rao [4].
(e) Code by Zhang and Xia [20], non constructable.
(*) Proposed code improving the existing code.

the construction can be easily implemented in hardware for information encoding and decoding. In Section 3.2, the construction method of the proposed codes is explained. In Section 3.3, some methods to improve new partitions, that are used in the construction method, are discussed. Then, in Section 3.4, the new single asymmetric error correcting codes are introduced. Finally, some useful remarks are given in Section 3.5.

3.2 A Construction Method for Single Asymmetric Error Correcting Codes

The proposed construction method is based on the Cartesian product of two sets of partitioned codes of smaller dimensions. Although the Cartesian product of two sets is well-known, it has a slightly different meaning when the two sets are codes.

Definition 3.1 Let X and Y be two sets. Then the *Cartesian product* of X and Y is defined as follows

$$X \times Y = \{(x, y) : \ x \in X \text{ and } y \in Y\}$$

Definition 3.2 The *Cartesian product of two codes*, X and Y, is the code $X \times Y$ such that every pair $(x, y) \in X \times Y$ is a code word which is the concatenation of the code word $x \in X$ and the code word $y \in Y$.

Example 3.1 Let $X = \{00, 11\}$ and $Y = \{001, 011, 111\}$ be two codes. The Cartesian product of X and Y is the code

$X \times Y = \{00001, 00011, 00111, 11001, 11011, 11111\}$.

Consider the first code word in $X \times Y$, which is 00001; clearly it is the concatenation of the code word $00 \in X$ and the code word $001 \in Y$. This is true for every code word in $X \times Y$.

According to Definition 3.2, if the code X is of length p and the code Y is of length q, then the code $X \times Y$ is of length $p + q$. This means that a code of larger length can be formed by the Cartesian product of two codes of smaller lengths. Before going further in explaining the construction method, consider the following definition:

Definition 3.3 Let X be a set of binary vectors, and let $X_1, X_2, \ldots, X_m$ be m subsets of X. The set $\{X_1, X_2, \ldots, X_m\}$ is called a *partition* of X if the following two conditions hold:

1. $X_i \cap X_j = \phi$ for $i \neq j$, and

2. $\bigcup_{i=1}^{m} X_i = X$.

The subsets $X_1, X_2, \ldots, X_m$ are called *classes*; and the set X is said to be *partitioned* into m classes.

The construction method of a single asymmetric error correcting code is based on the Cartesian product of two sets of partitioned codes of smaller dimensions, called A- and *B-partitions*. These partitions are defined as follows:

Definition 3.4 Let A be the set of all the 2^p binary vectors of length p and let $\{A_1, A_2, \ldots, A_{p'}\}$ be a partition of A, such that $\Delta(A_i) \geq 2$ for $1 \leq i \leq p'$. Then, the partition $\{A_1, A_2, \ldots, A_{p'}\}$ is called an *A-partition* of length p.

Definition 3.5 Let B be the set of the 2^{q-1} even weight binary vectors of length q and let $\{B_1, B_2, \ldots, B_{q'}\}$ be a partition of B such that $\Delta(B_j) \geq 2$ for $1 \leq j \leq q'$. Then, the partition $\{B_1, B_2, \ldots, B_{q'}\}$ is called a *B-partition* of length q.

To construct a single asymmetric error correcting code of length n, two sets of partitioned codes, namely: *A-partition* $= \{A_1, A_2, \ldots, A_{p'}\}$ and *B-partition* $= \{B_1, B_2, \ldots, B_{q'}\}$ defined as above, are involved. The lengths, say p and q, of these two partitions satisfy $p + q = n$. The constructed code, denoted by C_n, of length n is the union of the Cartesian products of all pairs $(A_i \times B_i)$ in these two partitioned codes, i.e.

$$C_n = A_1 \times B_1 \cup A_2 \times B_2 \cup A_3 \times B_3 \cup \cdots \cup A_t \times B_t \tag{3.1}$$

where $t = min\{p', q'\}$.

The cardinality of the code is clearly:

$$|C_n| = |A_1| * |B_1| + |A_2| * |B_2| + |A_3| * |B_3| + \cdots + |A_t| * |B_t| \tag{3.2}$$

Theorem 3.1 *The code C_n of length $n = p + q$, obtained in* Equation (3.1), *is a single asymmetric error correcting code.*

Proof: Let $x, y \in C_n$ and $x \neq y$. Let $x = x_A x_B$ and $y = y_A y_B$ where $x_A \in A_i$, $x_B \in B_i$, $y_A \in A_j$, and $y_B \in B_j$.

Case 1, $(i = j)$:

either $x_A \neq y_A \Rightarrow d_a(x_A, y_A) \geq 2 \Rightarrow d_a(x, y) \geq 2$

or $x_B \neq y_B \Rightarrow d_a(x_B, y_B) \geq 2 \Rightarrow d_a(x, y) \geq 2$.

Case 2, $(i \neq j)$:

Here we have $d_H(x_A, y_A) \geq 1$ since $x_A \neq y_A$, and $d_H(x_B, y_B) \geq 2$ since x_B and y_B are both of even weight and $x_B \neq y_B$. Therefore, $d_H(x, y) \geq 3 \Rightarrow d_a(x, y) \geq 2$.

□

Example 3.2 To construct a single asymmetric error correcting code C_6, let $p = 2$ and $q = 4$. Then $A = \{00, 01, 10, 11\}$ can be partitioned into $A_1 = \{00, 11\}$, $A_2 = \{01\}$, and $A_3 = \{10\}$; and $B = \{0000, 0011, 0101, \ldots, 1111\}$ can be partitioned into $B_1 = \{0000, 0011, 1100, 1111\}$, $B_2 = \{0101, 1010\}$, and $B_3 = \{0110, 1001\}$. We obtain a code C_6 of length $2 + 4 = 6$, where $C_6 = A_1 \times B_1 \cup A_2 \times B_2 \cup A_3 \times B_3$, having $2 * 4 + 1 * 2 + 1 * 2 = 12$ code words as shown in Figure 3.1.

From Equation (3.2), the size of the constructed code depends on the sizes of the partitions. In order to maximize the size of the code C_n of length n, appropriate

00	0000	
00	0011	
00	1100	
00	1111	$A_1 \times B_1$
11	0000	
11	0011	
11	1100	
11	1111	
01	0101	
01	1010	$A_2 \times B_2$
10	0110	
10	1001	$A_3 \times B_3$

Figure 3.1: Constructing a single asymmetric error correcting code for $n = 6$.

values of p and q, satisfying $n = p + q$, should be chosen. Without loss of generality, any partition $P = \{A_1, A_2, \ldots, A_m\}$ is assumed to satisfy $|A_j| \geq |A_{j+1}|$ for all j. Once p and q are chosen, "good" A- and *B-partitions* should be obtained. A partition $P_1 = \{A_1, A_2, \ldots, A_m\}$ is better than $P_2 = \{A'_1, A'_2, \ldots, A'_{m'}\}$ if $|A_j| \geq |A'_j|$ for all j such that $1 \leq j \leq m$. This is because P_1 in this case yields more code words than P_2 according to Equation (3.2).

In general, we may not be able to obtain the best partition (in fact, it may not exist), but as a rule of thumb, a good partition should have as few classes as possible and classes' size should be maximized. After the selection of the A- and *B-partitions*, the code is formed by applying the Cartesian product of the largest class of *A-partition* with the largest class of *B-partition*, then the second largest with the second largest and so on.

3.2.1 *A-Partitions*

The *A-partition* of a code of a given length can be obtained using the *Abelian group* partitioning, given by Varshamov in 1973 [17]. This method was improved and used again in 1979 by Constantin and Rao [4]. In this method, a code A of some length p is partitioned into $p+1$ disjoint sets, $A_1, A_2, \ldots, A_{p+1}$, such that $\Delta(A_i) \geq 2$ for $1 \leq i \leq p+1$.

For a set A of binary vectors of length p, the *group* partition (Γ_p) of $p+1$ classes is constructed as follows:

Algorithm: Group Patitioning

Input: set of binary vectors A

Output: the *A-partition* Γ_p

1. Initialize $\Gamma_p = \{A_1, A_2, \ldots, A_{p+1}\}$ where $A_i = \phi$ for all i.
2. For every code word $c = [c_1\ c_2\ \cdots\ c_p] \in A$, where $c_i \in \{0, 1\}$, do
   - compute the sum
   $$k = \left(\sum_{j=1}^{p} j \cdot c_j \right) \bmod (p+1)$$
   - add c to the class A_{k+1}.
3. Return Γ_p

End.

It is shown in [4] that this algorithm constructs $p+1$ classes of asymmetric distance greater than or equal to 2. Of course, the largest class is at least of size $2^p/(p+1)$. Table 3.2 shows the *A-partitions* obtained using this method.

p	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}
2	2	1	1								
3	2	2	2	2							
4	4	3	3	3	3						
5	6	5	5	6	5	5					
6	10	9	9	9	9	9	9				
7	16	16	16	16	16	16	16	16			
8	30	28	28	29	28	28	29	28	28		
9	52	51	51	51	51	52	51	51	51	51	
10	94	93	93	93	93	93	93	93	93	93	93

Table 3.2: *A-partitions* using the Abelian group partitioning (Γ_p).

The *group* partitions are used to obtain most of the *A-partitions* in order to construct the proposed codes in this chapter. In some cases, partitions that are better than *group* partitions can be obtained. For example, for $p = 5$, Table 3.3 gives a better partition obtained manually. Comparing this partition with the *group* partition Γ_5 in Table 3.4, the former has more classes of large sizes than the latter. Section 3.3.1 explains a procedure which can be used to obtain better *A-partitions* in many other cases, for example, for $p = 6, 10$, and 11. In Section 3.3.2, another method of partitioning using graph coloring techniques is explained. Table 3.5 shows the *A-partitions* used in this thesis to construct the proposed codes. These are the best known partitions. Of course, better partitons should be used whenever they are available.

A_1	A_2	A_3	A_4	A_5	A_6
00001	00010	00100	10000	01000	00000
11000	01100	10010	01010	10100	11100
00110	10001	01001	00101	00011	01011
10011	11010	10101	10110	10111	11111
01101	00111	01110	11001		
11110	11101	11011	01111		

Table 3.3: The six classes in the *A-partition* for $p = 5$.

3.2.2 *B-Partitions*

The *B-partitions* shown in Table 3.6 are obtained from the partitioning of the constant weight vectors into classes with Hamming distance 4 (see [3]). For example, the entries for $q = 4$ which are 4, 2, and 2 are obtained as follows: First, the vectors of weight 0 are partitioned into one class, namely {0000}; the vectors of weight 2 are partitioned into three classes: {0011, 1100}, {1001,0110}, and {1010,0101}; and the vectors of weight 4 have one class which is {1111}. The *eight* even weight vectors of length 4 can then be partitioned into three classes of size 4, 2, and 2 respectively as follows: {0000, 0011, 1100, 1111}, {1001,0110}, and {1010,0101}, where each class is of asymmetric distance 2. The constant weight partitions of different weights are listed in [3] for binary vectors of lengths up to 14. Partitions of larger even weight vectors can be obtained using the procedure given in [3], and partitions of different even weights can be assembled (as given in the above example) to obtain partitions

A_1	A_2	A_3	A_4	A_5	A_6
00000	10000	01000	11000	10100	01100
11100	11010	10110	00100	00010	10010
01010	00110	11001	01110	11110	00001
10001	01001	00101	10101	01101	11101
11011	10111	01111	00011	10011	01011
00111			11111		

Table 3.4: The six classes of the partition Γ_5.

of all even weight vectors of the required length.

It was shown in [15] that when $q = 2^i$, or $q = 3 \times 2^i$ for $i \geq 1$, the even weight vectors can be partitioned into $q - 1$ classes. For example, when $q = 4$, it gives a partition with *three* classes of the following sizes: $4, 2$ and 2. The procedure given in [15] also works when $q = 5 \times 2^i$, for $i \geq 1$; this is because, as given in Table 3.6, the even weight vectors of length 10 can be partitioned into *nine* classes, and consequently, when $q = 5 \times 2^i$, it can be partitioned into $5 \times 2^i - 1$ classes.

3.3 New Partitions

As it is given by Equation (3.2), the construction method is quite sensitive to the sizes of the smaller partitions. The better partitions used in this method, the more code words constructed. One way to obtain the *A-partitions* is by using the Abelian *group* partitioning explained in Section 3.2.1. However, these partitions can be im-

p	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	Remark
1	1	1											[17] Z_2
2	2	1	1										[17] Z_3
3	2	2	2	2									[17] Z_4
4	4	3	3	3	3								[17] Z_5
5	6	6	6	6	4	4							see Table 3.3
6	12	10	10	8	8	8	8						Section 3.3.1
7	18	18	18	18	17	16	13	10					Figure 3.8
8	32	28	28	28	28	28	28	28	28				[[4] $Z_3 \times Z_3$
9	52	52	51	51	51	51	51	51	51	51			[17] Z_{10}
10	104	102	102	102	102	90	88	84	84	84	82		Section 3.3.1
11	180	180	176	172	172	168	168	168	168	168	164	164	Section 3.3.1

Table 3.5: *A-partitions.*

q	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}
1	1										
2	2										
3	2	1	1								
4	4	2	2								
5	4	3	3	3	3						
6	8	6	6	6	6						
7	12	11	10	10	9	8	4				
8	24	22	20	20	18	16	8				
9	36	35	35	35	33	32	32	13	5		
10	72	70	70	70	62	60	54	40	14		
11	125	124	118	117	110	101	100	94	79	46	10
12	248	246	234	234	224	198	192	176	136	94	66

Table 3.6: *B-partitions.*

proved using the methods explained in Sections 3.3.1 and 3.3.2. Unlike *A-partitions*, the existing *B-partitions* are very tight, and it seems to be hard to get any significant improvement over there.

3.3.1 Improving *A-Partitions* from Smaller Sets of Partitions

Recall the construction method given in Section 3.2 for constructing single asymmetric error correcting codes. A procedure similar to this method can be used to improve *A-partitions*. The goal is to produce an *A-partition* of $p+1$ classes[1] which is better than the *group* partition for the same value of p.

In order to obtain a partition of all binary vectors of length p, two numbers s and t are chosen such that

$$s = \left\lfloor \frac{p-1}{2} \right\rfloor \tag{3.3}$$

and

$$t = p - s \tag{3.4}$$

which implies

$$t = \left\lceil \frac{p+1}{2} \right\rceil \tag{3.5}$$

[1]It may however be possible to find an *A-partition* with more than $p+1$ classes, yet it is more useful for constructing asymmetric error correcting codes than the one with $p+1$ classes.

Then, all classes in a partition of vectors of length s, and all classes in partitions of the odd as well as of the even weight vectors of length t are employed in different distinct combinations to produce the desired partitions of length p. It is always possible to get a partition with $s+1$ classes of the vectors of length s, and t classes of all odd (or even) weight vectors of length t; this is because the former is the same as the *A-partitions*, and the latter is similar to the *B-partitions*.

From Equation (3.5), it follows that

$$t = \begin{cases} (p+1)/2 & \text{if } p \text{ is odd} \\ (p+2)/2 & \text{if } p \text{ is even} \end{cases} \tag{3.6}$$

which implies that

$$2t = \begin{cases} p+1 & \text{if } p \text{ is odd} \\ p+2 & \text{if } p \text{ is even} \end{cases} \tag{3.7}$$

Therefore, it is always possible to obtain $2t$ classes of binary vectors of length p.

The proposed procedure produces better *A-partitions* (than the *group* partitions) for all odd values of p, but only for certain even values of p, as detailed below.

- When $p \bmod 4 = 0$, more classes are obtained using this procedure, so the $p+1$ classes obtained using the *group* method are better than these of this procedure, in this case.

- For odd values of p, i.e. when $p \bmod 4 = 1$ or 3, always $p+1$ classes can be obtained using this procedure, and in many cases it gives better partitions than

the *group* partitions. However, it is observed that this procedure produces a flat partition (that is, all classes having nearly equal number of elements), as does the *group* method, when $p = 2^i - 1$.

- Finally, when $p \bmod 4 = 2$, this procedure gives $p+1$ classes which are better than those obtained using the *group* method whenever $t = r \times 2^i$, where $1 \leq r \leq 6$: this is because, as described in Section 3.2.2, one can get $t - 1$ classes of the even weight vectors of length t in these cases.

Therefore, for all odd values of p, and for quite a few cases when p is even, this procedure produces *A-partitions* which are at least as good as (and in many cases better than) those obtained using the *group* method.

Example 3.3 In this example, the *A-partition* for $p = 6$ of seven classes of the sizes: 12, 10, 10, 8, 8, 8, and 8 is illustrated. Here, $s = \lfloor \frac{6-1}{2} \rfloor = 2$, and $t = \lceil \frac{6+1}{2} \rceil = 4$. Recall that one can partition all binary vectors of length 2, $S = \{00, 01, 10, 11\}$, into

$S_1 = \{00, 11\}$,

$S_2 = \{01\}$, and

$S_3 = \{10\}$.

The eight even weight binary vectors of length 4,

$T = \{0000, 0011, 0101, 0110, 1001, 1010, 1100, 1111\}$, can be partitioned into:

$T_1 = \{0000, 0011, 1100, 1111\}$,

$T_2 = \{0101, 1010\}$, and

$T_3 = \{0110, 1001\}$.

And the eight odd weight vectors.

$T' = \{0001, 0010, 0100, 0111, 1000, 1011, 1101, 1110\}$, can be partitioned into four classes:

$T'_1 = \{0001, 1110\}$,

$T'_2 = \{0010, 1101\}$,

$T'_3 = \{0100, 1011\}$, and

$T'_4 = \{1000, 0111\}$.

Now the seven classes of the *A-partition* of all the 2^6 binary vectors can be obtained as illustrated in Figure 3.2. Notice that $A_1 \cup A_2 \cup \ldots \cup A_7$ contain all the 64 binary vectors of length 6, $A_i \cap A_j = o$ when $i \neq j$, and $\Delta(A_i) \geq 2$ for $1 \leq i \leq 7$.

The sizes of these seven classes are: 12, 10, 10, 8, 8, 8, and 8, as given in Table 3.5. This can be contrasted with the flat partition, Γ_6, of the 2^6 binary vectors obtained using the *group* method, where the seven classes are of the sizes: 10, 9, 9, 9, 9, 9, and 9, as shown in Table 3.7.

This procedure may be deemed as a generalized version of the code construction method proposed in Section 3.2. Clearly, each A_i of length p in the above example is obtained in the same way C_p is obtained[2], only using different combinations of S, T and T' partitions. Theorem 3.1 shows that the code C_p obtained using

[2]Recall that C_p is a code of length p constructed by Equation (3.1).

$$A_1 = S_1 \times T_1 \cup S_2 \times T_2 \cup S_3 \times T_3 \text{ of size } 12$$

$$A_2 = S_1 \times T_2 \cup S_2 \times T_3 \cup S_3 \times T_1 \text{ of size } 10$$

$$A_3 = S_1 \times T_3 \cup S_2 \times T_1 \cup S_3 \times T_2 \text{ of size } 10$$

$$A_4 = S_1 \times T_1' \cup S_2 \times T_2' \cup S_3 \times T_3' \text{ of size } 8$$

$$A_5 = S_1 \times T_2' \cup S_2 \times T_3' \cup S_3 \times T_4' \text{ of size } 8$$

$$A_6 = S_1 \times T_3' \cup S_2 \times T_4' \cup S_3 \times T_1' \text{ of size } 8$$

$$A_7 = S_1 \times T_4' \cup S_2 \times T_1' \cup S_3 \times T_2' \text{ of size } 8$$

Figure 3.2: Constructing *A-partition* for $p = 6$.

this procedure produces a set of binary vectors satisfying the minimum asymmetric distance of 2. Therefore, it is possible to offer arguments similar to those made in the proof of Theorem 3.1 to show that each of the A_i's in the above procedure produces partitions satisfying the minimum asymmetric distance property. Notice that each of the A_i's employs a unique combination of S, T and T' partitions, and that $\sum |A_i| = 2^p$. Thus, it can be concluded that the proposed partitioning method produces valid partitions of the 2^p binary vectors.

It is not necessary to always match the even classes with rotated versions of the other classes. For instance, if there are four classes in each of S, T and T', then the combination given in Figure 3.3 would give *A-partitions* better than those obtained using the simple rotation strategy used in the previous example. It is observed that

A_1	A_2	A_3	A_4	A_5	A_6	A_7
000000	100000	010000	110000	101000	011000	111000
110100	101100	011100	001000	000100	100100	010100
001100	110010	101010	111100	111010	000010	100010
010010	001010	000110	011010	010110	110110	101110
011110	111110	110001	100110	011001	001110	000001
100001	010001	001001	101001	100101	111001	110101
101101	011101	111101	000101	000011	010101	001101
110011	101011	011011	111011	110111	100011	010011
001011	000111	100111	010111	001111	101111	011111
111111						

Table 3.7: The seven classes of the partition Γ_6.

the "best" strategy depends on the lengths and distributions of S, T and T'.

3.3.2 Improving *A-Partitions* Using Graph Coloring

In this section, graph-coloring techniques are used to construct *A-partitions*. To construct an *A-partition* of length p, a graph $G = (V, E)$ of 2^p nodes, where V is the set of the nodes and E is the set of all edges in the graph, is constructed as follows:

V is the set of all the 2^p binary vectors and

$E = \{(x, y) : x, y \in V \text{ and } d_a(x, y) = 1\}$

The nodes of the graph are colored using m colors, such that $\forall x, y \in V$, if $(x, y) \in E$ then x and y have different colors. Therefore, the set of all nodes having color k, say A_k, satisfies $\Delta(A_k) \geq 2$.

$$A_1 = S_1 \times T_1 \cup S_2 \times T_2 \cup S_3 \times T_3 \cup S_4 \times T_4$$

$$A_2 = S_1 \times T_2 \cup S_2 \times T_1 \cup S_3 \times T_4 \cup S_4 \times T_3$$

$$A_3 = S_1 \times T_3 \cup S_2 \times T_4 \cup S_3 \times T_1 \cup S_4 \times T_2$$

$$A_4 = S_1 \times T_4 \cup S_2 \times T_3 \cup S_3 \times T_2 \cup S_4 \times T_1$$

$$A_5 = S_1 \times T_1' \cup S_2 \times T_2' \cup S_3 \times T_3' \cup S_4 \times T_4'$$

$$A_6 = S_1 \times T_2' \cup S_2 \times T_1' \cup S_3 \times T_4' \cup S_4 \times T_3'$$

$$A_7 = S_1 \times T_3' \cup S_2 \times T_4' \cup S_3 \times T_1' \cup S_4 \times T_2'$$

$$A_8 = S_1 \times T_4' \cup S_2 \times T_3' \cup S_3 \times T_2' \cup S_4 \times T_1'$$

Figure 3.3: A combination of S, T, and T' to construct an *A-partition* without rotation strategy.

The following example shows how graph-coloring can be used to partition a set of binary vectors.

Example 3.4 In this example, graph-coloring is used to partition the set $V = \{0011, 0110, 1100, 0001, 0000, 1001\}$. First, construct the graph $G = (V, E)$ whose nodes are the elements of V, and it has an edge between x and $y \in V$ if and only if $d_a(x, y) = 1$. The constructed graph is illustrated in Figure 3.4. It has the following set of edges:

$E = \{(0011, 0110), (0011, 0001), (0011, 1001),$

$(1100, 0110), (1100, 1001), (0000, 0001), (0001, 1001)\}$.

This graph can be colored with three different colors using first-fit algorithm such that there is no two adjacent nodes having the same color. Figure 3.5. The coloring assignment partitions the set V into three disjoint subsets as follows (Figure 3.6):

$V_1 = \{0011, 1100, 0000\}$,

$V_2 = \{0110, 1000\}$, and

$V_3 = \{1001\}$.

Notice that $\Delta(V_i) \geq 2$, for $i = 1, 2, 3$.

In order to obtain the *A-partitions* that can be used in the Cartesian product method given in Section 3.2, all the 2^p code words should be included in the set V, where p is the length of the *A-partitions*. Then, V is partitioned into some subsets of minimum asymmetric distances $\Delta \geq 2$ using graph-coloring method. Coloring a graph of 2^p nodes is not an easy task since it is an NP-complete problem. So, the statement of coloring problem can be modified slightly and the following version of coloring problem maybe considered instead:

Problem: *"Given a graph $G = (V, E)$ and a fixed number of colors m, find some large subset V' such that V' is $m-$colorable and $V' \subseteq V$."*

So, the goal here is not to color all the nodes, but to color as many nodes as possible. Hence, some nodes can be skipped in order to go further in the coloring algorithm.

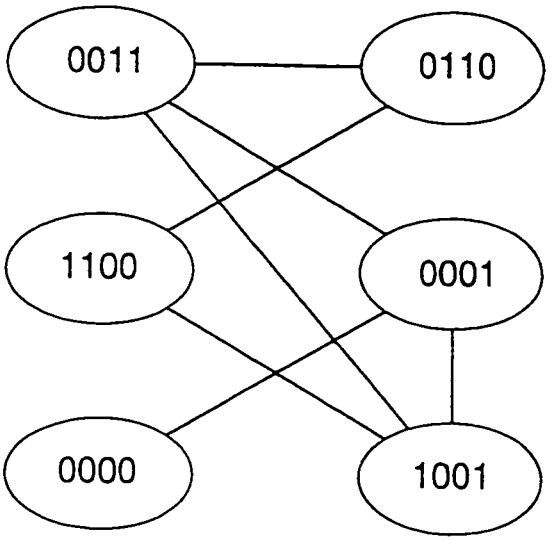


Figure 3.4: Constructing the graph G for the coloring example.

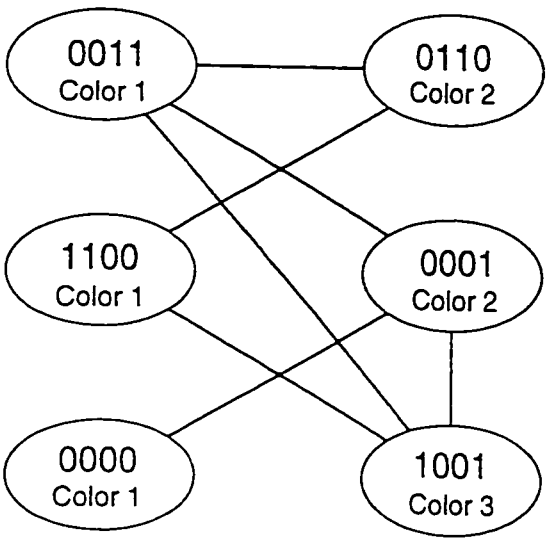


Figure 3.5: Coloring the graph G using three colors.

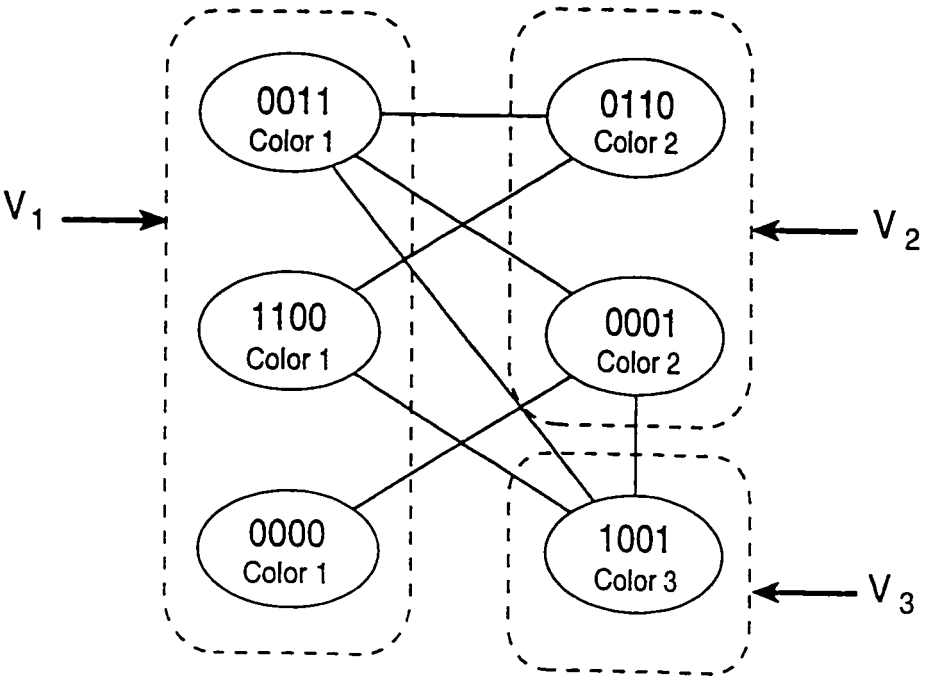


Figure 3.6: Partitioning the set V using graph coloring.

In the following, we present the modified coloring algorithm called *Coloring In-Limited-Backtracking Algorithm* (CILBA). The pseudocode of this algorithm is given below. In CILBA, some heuristics have been used to enhance its performance.

To find an *A-partition* of length p, CILBA colors a graph $G = (V, E)$ of 2^p nodes. Let $V = \{x_1, x_2, \ldots, x_n\}$, where $n = 2^p$, be the set of nodes in G. The heuristics start by skipping the first k nodes of the highest degrees. Then, the remaining nodes are sorted in an ascending order according to their weights, i.e. $w(x_i) \leq w(x_{i+1})$. For x_i in this order, i is called the *depth* of node x_i. Next, exhaustive search is applied to color G with the following exception: When it reaches a certain maximum depth, CILBA should not backtrack for more than a limited *range* which is previously fixed for every depth. When it fails to backtrack, CILBA loads the last status saved for the maximum depth achieved, and skips the current node (x_i) that cannot be colored using m colors or may need a "further" backtracking. The pseudocode of CILBA is given in the following:

Algorithm: CILBA

INPUT: $G = (V, E), m, k$

$V = \{x_1, x_2, \ldots, x_n\}$

$E = \{(x, y) : x, y \in V \text{ and } d_a(x, y) = 1\}$

m = number of colors

k = number of nodes to be skipped at the beginning

OUTPUT: $Status, MaxDepth$

Let $n = |V|$;

$\forall x \in V$, initialize:

$Color[x]$ = 0;

$deg[x]$ = degree of x, i.e. the number of edges in x;

$w[x]$ = number of 1's in the binary vector x;

$Uncolorable[x]$ = False;

Sort all $x \in V$ such that $deg[x_i] \geq deg[x_{i+1}]$;

Sort all x_i for $i = k+1$ to n such that $w[x_i] \leq w[x_{i+1}]$;

For $i = k+1$ to n, Initialize $Range[i]$;

$i = k+1$;

$Color[x_i] = 1$;

$MaxDepth = i$;

Save $Status$;

$i = i+1$;

while $i \leq n$ and $i > k+1$ do

    for $j = Color[x_i]+1$ to m do

        if x_i is colorable by j then

            $Color[x_i] = j$;

            if $MaxDepth < i$ then

                $MaxDepth = i$;

                Save $Status$

```
            i = i + 1;
            Exit-for-loop;
        end-for
        if x_i is not colored by j then
            if MaxDepth − i < Range[MaxDepth] then  /* can backtrack */
                Color[x_i] = 0;
                i = i − 1;              /* backtrack to the previous node */
            else                                    /* cannot backtrack */
                Load Status;
                i = maxdepth + 1;
                Mark x_i Uncolorable;
                i = i + 1;                  /* consider the next node */
            end-if
    end-while
    Return Status and maxdepth;
    End.
```

The *Range* in CILBA controls the performance of the algorithm. For a fixed number of colors (m), small ranges normally yield a fast execution but with many *uncolorable* nodes. while large ranges make CILBA slower but more accurate. Thus, CILBA colors more nodes with larger ranges for a fixed number of colors.

Moreover, CILBA can simulate both the *first-fit* and the *exhaustive search* coloring algorithms. To simulate the *first-fit* algorithm, all the ranges are set to zero, and m is set to *infinity*. In this case CILBA never backtracks since there are infinitly suffecient colors. For simulating the *exhaustive search* coloring algorithm, the ranges are set to *infinity*. Hence, CILBA may backtrack from any depth to the node at depth 1 regardless of the current maximum depth achieved.

Since nodes[3] are sorted according to their weights and there are no edges between nodes of weight w and any node of weight $w \pm 2$, a "reasonable" range for a node x_i of weight w at depth i is computed as:

$$Range[x_i] = i - e \tag{3.8}$$

where e is the minimum depth of all nodes of weight $w-1$, as shown in Figure 3.7. Equation (3.8) can be expressed as follows:

$$Range[x_i] = i - \sum_{j=0}^{w-2} |S_j| - 1 \tag{3.9}$$

where S_j is the set of all nodes of weight j. Of course, smaller ranges can be used instead to make CILBA faster; however, this may reduce the accuracy of the algorithm. Using ranges as given by Equation(3.9), CILBA indeed was used to construct a new *A-partition* for $p = 7$ (Figure 3.8). It gives much better partition for $p = 7$ than the *Abelian group* partition, Γ_7, which is very flat (i.e all the classes

[3]Recall that nodes in the graph are code words, and there is no edges between node x and node y if $d_a(x, y) \geq 2$.

have the same size), as shown in Figure 3.9. The new *A-partition* is very useful for the construction method given by Equation (3.1). It is used to construct the proposed codes of lengths 15, 17 and 19 with the *B-partitions* for $q = 8$, 10 and 12 respectively. The sizes of the proposed codes are given in Section 3.4.

3.4 The New Codes

The new single asymmetric error correcting codes proposed here are the results of applying the Cartesian product method given by Equation (3.1). The sizes of these codes are given by Equation (3.2). Using the same method, we have published in [1] most of the codes proposed here (Table 3.8). However, the *A-partition* for $p = 7$ used in [1] is Γ_7, which is obtained from the Abelian *group* partitioning (see Figure 3.9). Since that one is a flat partition, i.e. all the A_i's are of the same size, which is 16, the Cartesian product method gives poor results. In [10], some graph-coloring techniques were used to construct a better partition for $p = 7$. This yields a better code for $n = 19$ with 110 code words more than the one given in [1].

In this thesis, The new coloring algorithm, CILBA, has been used to find even better *A-partition* for $p = 7$ (Figure 3.8). This yields some improvements over the codes given in [1] and [10]. Comparing Table 3.1 with Table 3.8, the codes of lengths 15, 17 and 19 have been improved. These improvements were mainly due to the use of the new *A-partition* for $p = 7$ (Figure 3.8). The sizes of the *A-partitions* used

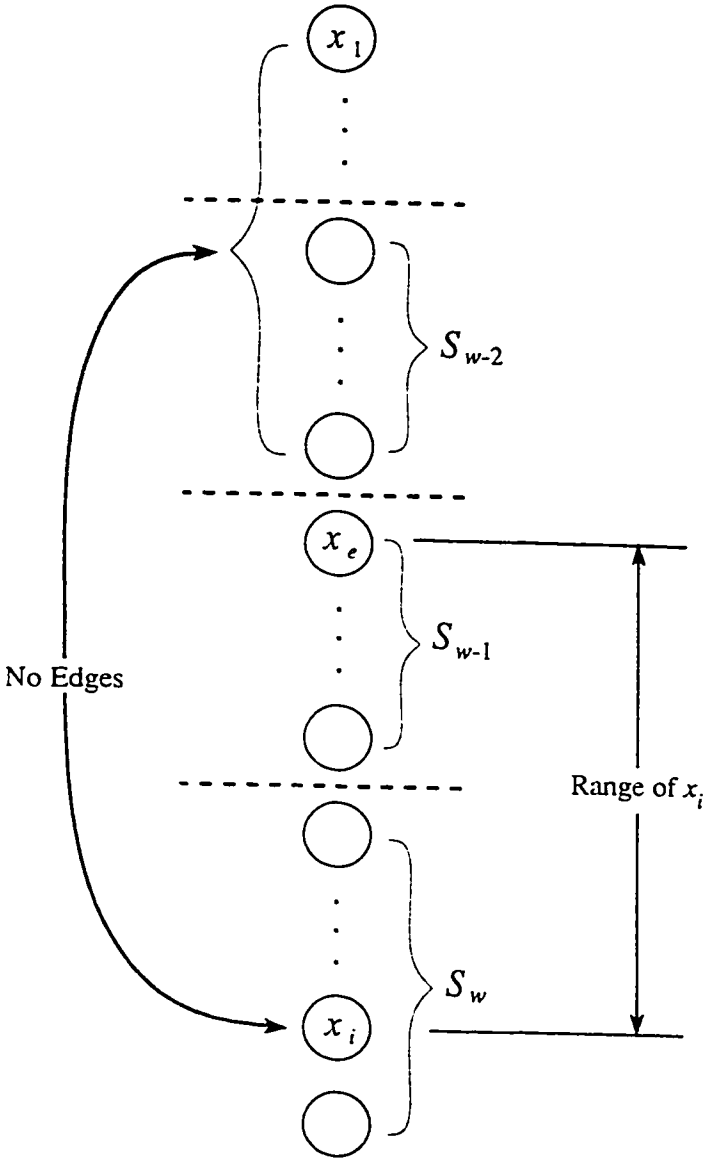


Figure 3.7: A reasonable range for x_i in CILBA.

A_1	A_2	A_3	A_4
0 0 1 0 0 0 0	0 1 0 0 0 0 0	0 0 0 0 1 0 0	0 0 0 0 0 0 0
0 0 0 0 0 1 1	0 0 0 1 0 1 0	0 0 0 1 0 0 1	0 0 1 1 0 0 0
0 0 0 1 1 0 0	0 0 1 0 0 0 1	0 1 0 0 0 1 0	0 1 0 0 1 0 0
1 1 0 0 0 0 0	1 0 0 0 1 0 0	1 0 1 0 0 0 0	1 0 0 0 0 0 1
0 0 1 1 0 0 1	0 0 1 0 1 1 0	0 0 0 1 1 1 0	0 0 0 0 1 1 1
0 1 0 0 1 0 1	0 1 0 0 0 1 1	0 0 1 0 1 0 1	0 1 0 1 0 1 0
0 1 1 0 0 1 0	0 1 0 1 1 0 0	0 1 1 1 0 0 0	0 1 1 0 0 0 1
1 0 0 1 0 1 0	1 0 0 1 0 0 1	1 0 0 0 0 1 1	1 0 0 1 1 0 0
1 0 1 0 1 0 0	1 1 1 0 0 0 0	1 1 0 0 1 0 0	1 0 1 0 0 1 0
0 0 1 0 1 1 1	0 0 1 1 1 0 1	0 1 0 1 1 0 1	0 0 1 1 0 1 1
0 1 0 1 1 1 0	0 1 1 1 0 1 0	0 1 1 0 0 1 1	0 1 1 1 1 0 0
1 0 0 1 1 0 1	1 0 0 1 1 1 0	1 0 1 0 1 1 0	1 0 1 0 1 0 1
1 1 0 0 0 1 1	1 0 1 0 0 1 1	1 0 1 1 0 0 1	1 1 0 0 1 1 0
1 1 1 1 0 0 0	1 1 0 0 1 0 1	1 1 0 1 0 1 0	1 1 0 1 0 0 1
0 1 1 1 1 0 1	0 1 1 0 1 1 1	0 1 1 1 1 1 0	0 1 0 1 1 1 1
1 0 1 1 0 1 1	1 1 0 1 0 1 1	1 0 0 1 1 1 1	1 0 1 1 1 1 0
1 1 1 0 1 1 0	1 1 1 1 1 0 0	1 1 1 0 1 0 1	1 1 1 0 0 1 1
1 1 0 1 1 1 1	1 0 1 1 1 1 1	1 1 1 1 0 1 1	1 1 1 1 1 0 1

A_5	A_6	A_7	A_8
0 0 0 0 0 1 0	0 0 0 1 0 0 0	0 0 0 0 0 0 1	1 0 0 0 0 0 0
0 0 1 0 1 0 0	1 0 0 0 0 1 0	0 0 0 0 1 1 0	0 0 0 0 1 0 1
0 1 0 0 0 0 1	0 1 1 0 0 0 0	1 0 0 1 0 0 0	0 0 1 0 0 1 0
0 0 0 1 1 0 1	0 0 1 0 0 1 1	1 1 0 0 0 1 0	0 1 0 1 0 0 0
0 0 1 1 0 1 0	0 0 1 1 1 0 0	0 0 0 1 0 1 1	1 0 0 0 1 1 0
0 1 0 0 1 1 0	0 1 0 1 0 0 1	0 1 1 0 1 0 0	1 0 1 1 0 0 0
1 0 1 0 0 0 1	1 0 0 0 1 0 1	0 0 1 1 1 1 0	1 1 0 0 0 0 1
1 1 0 1 0 0 0	0 0 0 1 1 1 1	0 1 0 0 1 1 1	1 0 0 1 0 1 1
0 1 0 1 0 1 1	0 1 1 0 1 1 0	0 1 1 1 0 0 1	1 1 1 0 1 0 0
0 1 1 0 1 0 1	1 0 1 1 0 1 0	1 0 1 0 1 1 1	1 1 1 1 1 1 1
1 0 0 0 1 1 1	1 1 0 1 1 0 0	1 1 0 1 1 0 1	
1 0 1 1 1 0 0	1 1 1 0 0 0 1	1 1 1 1 0 1 0	
1 1 1 0 0 1 0	0 1 1 1 0 1 1	0 1 1 1 1 1 1	
0 0 1 1 1 1 1	1 0 1 1 1 0 1		
1 1 0 1 1 1 0	1 1 0 0 1 1 1		
1 1 1 1 0 0 1	1 1 1 1 1 1 0		
1 1 1 0 1 1 1			

Figure 3.8: The new *A-partition* of the 2^7 binary vectors obtained using CILBA.

A_1	A_2	A_3	A_4
0000000	1000000	0100000	1100000
1011000	0111000	1111000	0010000
1100100	1010100	0110100	1110100
0010100	0001100	1001100	0101100
0100010	1100010	1010010	0110010
1111010	0010010	0001010	1001010
0110110	1110110	1101110	0000110
1001110	0101110	0011110	1011110
1000001	0100001	1100001	1010001
0111001	1111001	0010001	0001001
1010101	0110101	1110101	1101101
0001101	1001101	0101101	0011101
1100011	1010011	0110011	1110011
0010011	0001011	1001011	0101011
1110111	1101111	0000111	1000111
0101111	0011111	1011111	0111111

A_5	A_6	A_7	A_8
1010000	0110000	1110000	1101000
0001000	1001000	0101000	0011000
1101100	0000100	1000100	0100100
0011100	1011100	0111100	1111100
1110010	1101010	0000010	1000010
0101010	0011010	1011010	0111010
1000110	0100110	1100110	1010110
0111110	1111110	0010110	0001110
0110001	1110001	1101001	0000001
1001001	0101001	0011001	1011001
0000101	1000101	0100101	1100101
1011101	0111101	1111101	0010101
1101011	0000011	1000011	0100011
0011011	1011011	0111011	1111011
0100111	1100111	1010111	0110111
1111111	0010111	0001111	1001111

Figure 3.9: The eight classes of the partition Γ_7.
All the classes are of size 16 code words.

n	$\lvert C \rvert$
2	2
3	2
4	4
5	6
6	12
7	16
8	28
9	52
10	104
11	180
12	336
13	652
14	1204
15	2188
16	4232
17	7968
18	14624
19	28032
20	53856
21	101576
22	195700

Table 3.8: Single asymmetric error correcting codes.

to construct the proposed codes are listed in Table 3.5. The last column shows the reference and the method used to construct the partitions.

Table 3.9 shows the sizes of the proposed codes and the sizes of the known codes. The sizes of the proposed codes are also compared with $\lfloor \frac{2^n}{n} \rfloor$ and $\lfloor \frac{2^n}{n-1} \rfloor$. In many cases, the sizes of new single asymmetric error correcting codes satisfy $\lfloor \frac{2^n}{n} \rfloor \leq |C_n| \leq \lfloor \frac{2^n}{n-1} \rfloor$ for $2 \leq n \leq 22$. For $n = 11$ and 12, the lower bounds are not satisfied as shown in Table 3.9. This table also shows the values of p and q (chosen for the A- and *B-partitions* respectively) which are used in the Cartesian product method to construct a code of length $p+q$. All the *A-partitions* used here are taken from Table 3.5, while the *B-partitions* are from Table 3.6.

3.5 Remarks

As it is given by Equation (3.1), the construction method is procedural and can be implemented easily. The sizes of the proposed code, given by Equation (3.2), depend on the sizes of the A- and *B-partitions* involved. Obviously, better A- and *B-partitions* yield better codes. The construction method can be used recursively to construct better *A-partitions* from smaller partitions. Also, graph-coloring techniques can be used with some heuristics, like in CILBA, to obtain better *A-partitions*. Although this method is used to construct single asymmetric error correcting codes, a similar technique can also be used to construct codes capable of correcting more

n	$\lfloor \frac{2^n}{n} \rfloor$	existing code	p	q	proposed code	$\lfloor \frac{2^n}{n-1} \rfloor$	upper bound
2	2	2^a	0	2	2	4	2
3	2	2^a	1	2	2	4	2
4	2	4^a	0	4	4	5	4
5	6	6^a	1	4	6	8	6
6	10	12^b	2	4	12	12	12
7	18	18^d	3	4	16	21	18
8	32	36^c	2	6	28	36	36
9	56	62^d	3	6	52	64	62
10	102	108^c	4	6	104	113	117
11	186	174^c	5	6	180^*	204	210
12	341	340^e	4	8	336	372	410
13	630	624^e	5	8	652^*	682	786
14	1170	1139^e	6	8	1204^*	1260	1500
15	2184	2216^e	7	8	2214	2340	2828
16	4096	4168^e	6	10	4232^*	4369	5430
17	7710	7688^e	7	10	8192^*	8192	10379
18	14563	13951^e	8	10	14624^*	15420	19898
19	27594	26265^e	7	12	28548^*	29127	38008
20	52428	49940^a	8	12	53856^*	55188	73174
21	99864	95326^a	9	12	101576^*	104857	140798
22	190650	182326^a	10	12	195700^*	199728	271953

Table 3.9: Cardinalities of the new codes.

(a) Code by Varshamov [17].
(b) Code by Kim and Freiman [6].
(c) Code by Delsare and Piret [5].
(d) Code by Constantin and Rao [4].
(e) Code by Zhang and Xia [20], non constructable.
(*) Proposed code improving the existing code.

than one asymmetric error.

The proposed codes improve the existing lower bounds for all codes of length $n > 10$ (except $n = 12$ and $n = 15$). The proposed codes for $n = 12$ and $n = 15$ are still the best known codes that can be explicitly constructed, since the codes given in [20] are based on combinatorial arguments. It has been shown that, in many cases, the proposed codes contain at least $\lceil \frac{2^n}{n} \rceil$ code words, where n is the length of the code. Although the presented codes here are only up to $n = 22$, the construction procedures can be applied to codes with larger dimensions. In addition, whenever a value used for q which can be partitioned into $q - 1$ classes, one can get $|C_n| > \frac{2^n}{n}$ not only for $n = 2q - 2$, but also for other values of n in this vicinity where "good" p classes exist. For example, one can get 15 classes of even weight vectors when $q = 16$, and so he can obtain codes with size more than $\lceil \frac{2^n}{n} \rceil$, not only when $n = 30$, but for $n = 28$ and 29 as well.

An important remark here is on the size of the code C_{17}. The proposed code for $n = 17$ is of size $|C_{17}| = 8192 = 2^{13}$. This means that 13 bits of information (rather than 12) can be transmitted using 17-bit code words in a single asymmetric error correcting code.

For a given value of n, appropriate values of p and q must be chosen in order to maximize the code size. The values of p and q that produce codes for $n \leq 22$ are listed in Table 3.9. Notice that, in general, $p < q \leq p + 5$. Furthermore, for most of the values of n discussed in this chapter, q is chosen to be even and p is the largest

integer less than q: the only exception occurs when $n = 19$.

Finally, it was noticed that in many cases the single asymmetric error correcting codes satisfy $\lfloor \frac{2^n}{n} \rfloor \leq |C| \leq \lfloor \frac{2^n}{n-1} \rfloor$ for $2 \leq n \leq 22$. Only for $n = 11$ and 12, the lower bound is not satisfied as shown in Table 3.9. The codes have to be slightly improved for $n = 11$ and 12 to satisfy the lower bound. The existing upper bounds are more than $\lfloor \frac{2^n}{n-1} \rfloor$ for $n \geq 11$, so the observation about the upper bound may not be true for $n \geq 10$, but it is plausible.

Chapter 4

Conclusion and Future Work

4.1 Conclusion

In this thesis, new codes of asymmetric distance 2, capable of correcting a single asymmetric error, are introduced. The issue of asymmetric errors is relatively new, compared with the symmetric errors. However, many papers have been published in the area of asymmetric errors since the late 1950's due to their increasing number of applications. Examples include: transmissions in optical fibers, LSI single-transistor-cell memories, and MNOS memories.

The construction method of the proposed codes is also presented. This method is based on the Cartesian product of two sets of partitioned codes, say A- and *B-partitions*, of smaller dimensions. The method is quite sensitive to the sizes of the smaller partitions. The better partitions used in this method, the more code

words constructed. The issue of improving the *A-partitions* is discussed. Using the improved partitions, new single asymmetric error correcting codes of sizes larger than the existing ones are constructed.

It is worth noting that a code of 8192 code words of length 17 is obtained here, which is C_{17}. It is not only a power of 2 (i.e. $|C_{17}| = 2^{13}$ code words), but also it achieves the bound ($\frac{2^n}{n-1}$), which was only known for a code of length $n = 6$ with 12 code words. Using the code C_{17}, one can encode the 13-bit messages into code words of length 17 to form a code capable of correcting a single asymmetric error.

4.2 Future Work

There are several promising research directions that can be pursued based on the results of this thesis. The followings summarize some interesting directions for future work:

1. Using the proposed construction method to construct codes for correcting k asymmetric errors.

2. Designing algorithms for efficient encoding/decoding of the proposed codes.

3. Using the new graph-coloring algorithm (CILBA) to find better *A-partitions* for some other values of p.

4. Enhancing CILBA to avoid backtracking from node x to node y if there is no edge between x and y.

5. Finding upper bounds for single asymmetric error correcting codes tighter than the existing ones.

Bibliography

[1] S. Al-Bassam, R. Venkatesan, and S. Al-Muhammadi, "New single asymmetric error correcting codes," *IEEE Trans. Inform. Theory*, vol. IT-43, pp. 1619-1623, Sep. 1997.

[2] E. R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill, New York, USA, 1968.

[3] A. E. Brouwer, J. B. Shearer, N. J. A. Sloane, and W. D. Smith, "A new table of constant weight codes," *IEEE Trans. Inform. Theory*, Vol. 36, pp. 1334-1380, Nov. 1990.

[4] S. D. Constantin and T. R. N. Rao, "On the theory of binary asymmetric error-correcting codes," *Inform. Contr.*, Vol. 40, pp. 20-36, 1979.

[5] P. Delsarte and P. Piret, "Bound and constructions for binary asymmetric error-correcting codes," *IEEE Trans. Inform. Theory*, Vol. IT-27, pp. 125-128, Jan. 1981.

[6] W. H. Kim and C. V. Freiman, "Single error-correcting codes for asymmetric binary channels," *IRE Trans. Inform. Theory*, pp. 62-66, Jun. 1959.

[7] T. Kløve, "Uper bounds on codes correcting asymmetric errors," *IEEE Trans. Inform. Theory*, Vol. IT-27, pp. 128-131, Jan. 1981.

[8] S. Lin, *An Introduction to Error-Correcting Codes*, Prentice Hall, Englewood Cliffs, N.J., USA, 1970.

[9] F. J. MacWilliams and N. J. Sloane, *The Theory of Error-Correcting Codes*, Elsevier Science Publishers B.V., Amsterdam, Netherlands, 1977.

[10] S. Al-Muhammadi and S. Al-Bassam, "A new single asymmetric error correcting code of length 19," *Canadian Conference on Electrical and Computer Engineering*, vol. 1, pp. 75-77, May 1997.

[11] W. W. Peterson and E. J. Weldon, *Error-Correcting Codes*, MIT Press, Cambridge, Mass., USA, 1971.

[12] D. K. Pradhan (Editor), *Fault-Tolerant Computing: Theory and Techniques*, Chapter 4, "Coding theory for fault-tolerant systems," B. Bose and J. Metzner, pp. 265-335, Prentice Hall, Englewood Cliffs, N.J., USA, 1986.

[13] T. R. N. Rao and A. S. Chawla, "Asymmetric error codes for some LSI semiconductor memories," in *Proc. Ann. Southeastern Symp. Syst. Theory*, pp. 170-171, 1975.

[14] T. R. N. Rao and E. Fujiwara, *Error-Control Coding for Computer Systems*. Prentice Hall, Inc., Englewood Cliffs, N.J., USA, 1989.

[15] C. Van Pul and T. Etzion, "New lower bounds for constant weight codes," *IEEE Trans. Inform. Theory*, Vol. 35, pp. 1324-1329, Nov. 1989.

[16] R. R. Varshmov, "Estimate of the number of signals in codes with correction of nonsymmetric errors," *Automat. Telemekh.*, Vol. 25, pp. 1628-1629, 1964 (translation: *Automat. Rem. Contr.*), Vol. 25, pp. 1628-1629, 1964.

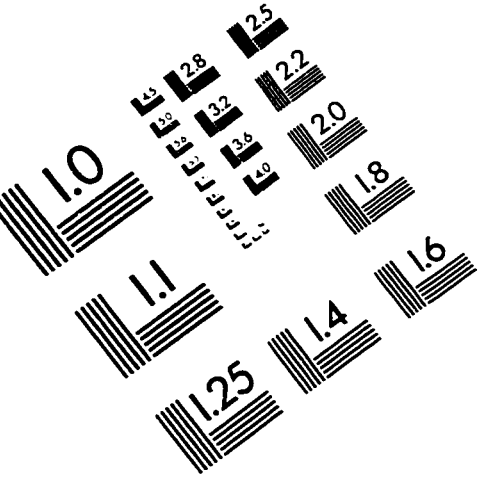
[17] R. R. Varshamov, "A class of codes for asymmetric channels and a problem from the additive theory of numbers," *IEEE Trans. Inform. Theory*, Vol. IT-19, pp. 92-95, Jan. 1973.

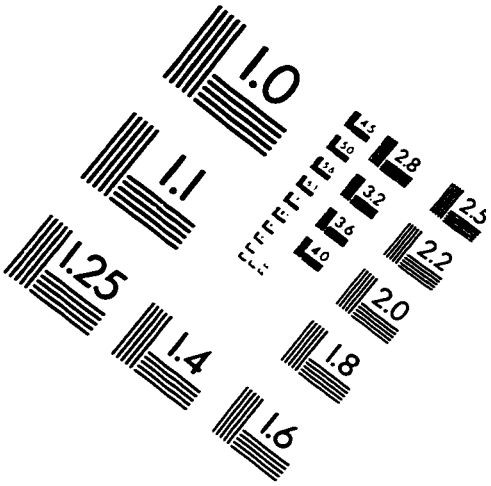
[18] J. Weber, C. de Vroedt, and D. Boekee, "New upper bounds on the size of codes correcting assymmetric errors," *IEEE Trans. Inform. Theory*, Vol. IT-33, pp. 434-437, May 1987.

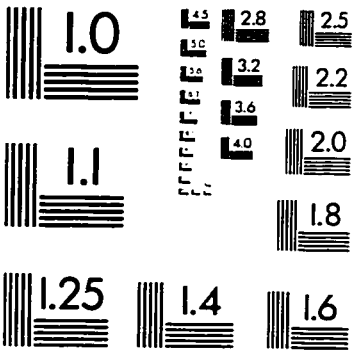
[19] J. Weber, C. de Vroedt, and D. Boekee, "Bounds and constructions for binary codes of length less than 24 and asymmetric distance less than 6," *IEEE Trans. Inform. Theory*, Vol. IT-34, pp. 1321-1331, Sep. 1988.

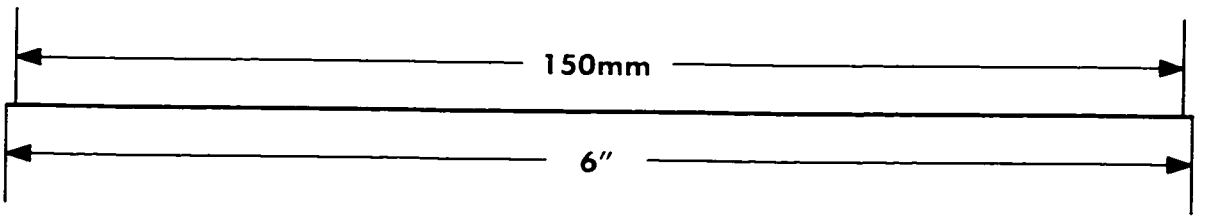
[20] Z. Zhang and X. Xia, "New lower bounds for binary codes of assymmetric distance two," *IEEE Trans. Inform. Theory*, Vol. IT-38, pp. 1592-1597, Sep. 1992.

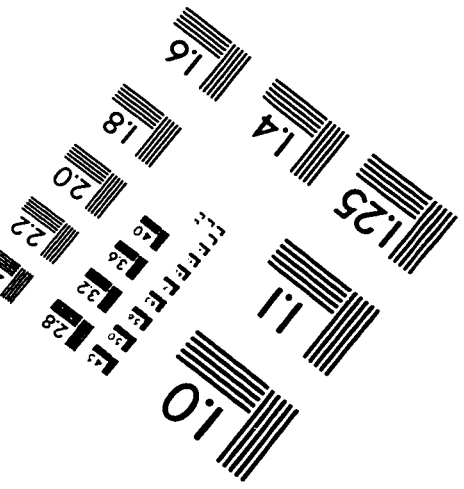
IMAGE EVALUATION TEST TARGET (QA-3)

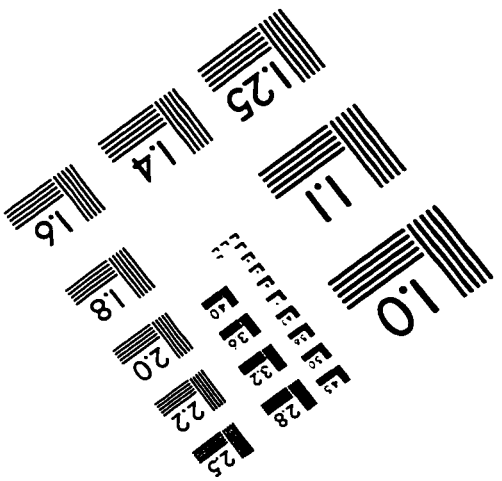












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989